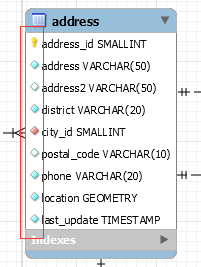
实验一报告

# 回答问题

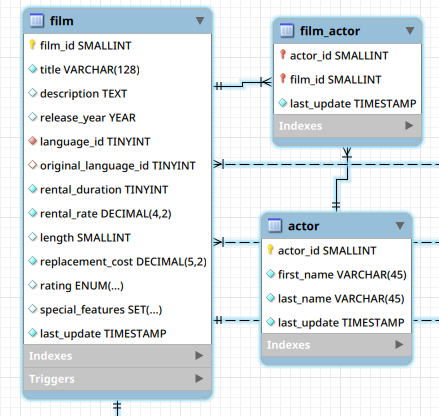
请一边熟悉sakila数据库，一边回答以下问题：

1. sakila.mwb模型中，表结构里每个字段前面的小标记分别表示什么意思？ （观察字段的属性）

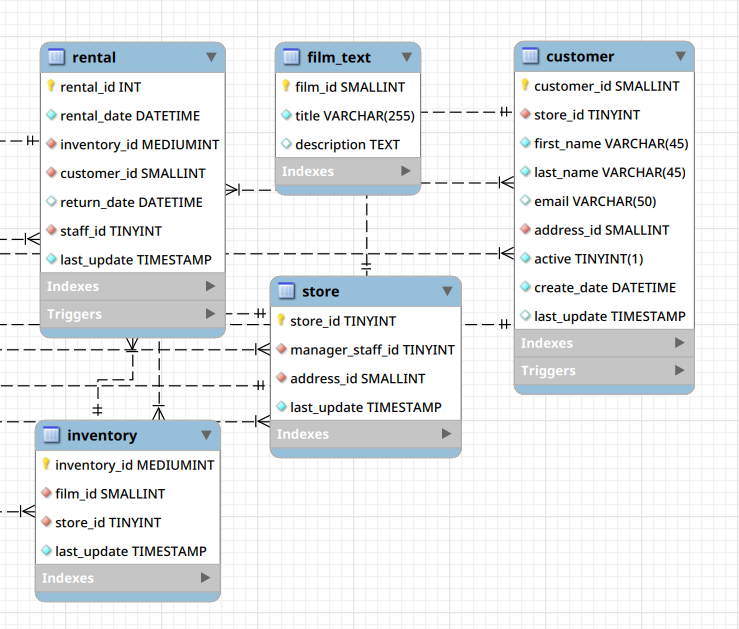


|  |  |
| --- | --- |
| 标记 | 意义 |
|  | 主键 |
|  | 非空的键 |
|  | 默认为空的键 |
|  | 来自其他表的非空外键 |

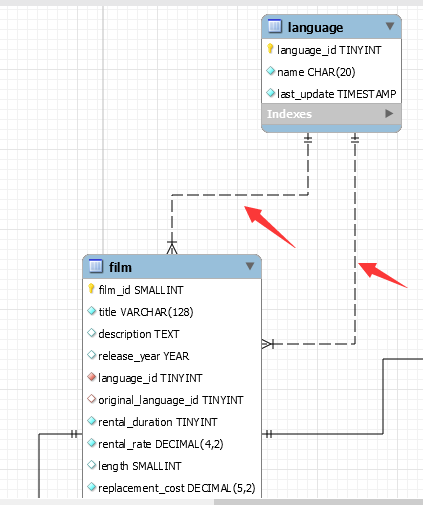
2. 图中哪部分体现影片-演员关系？换句话说，如果要找出演某个影片的演员名字，访问哪几张表可以获得信息？

这三个表体现了影片-演员关系：film、film\_actor、actor

3. 如果已知某个顾客姓名，要找到他租借的所有影片名，需要访问哪几张表？

需要访问 rental、film\_text、custormer、inventory、store这几张表

4. film和language表间的2条虚线表示什么意思？

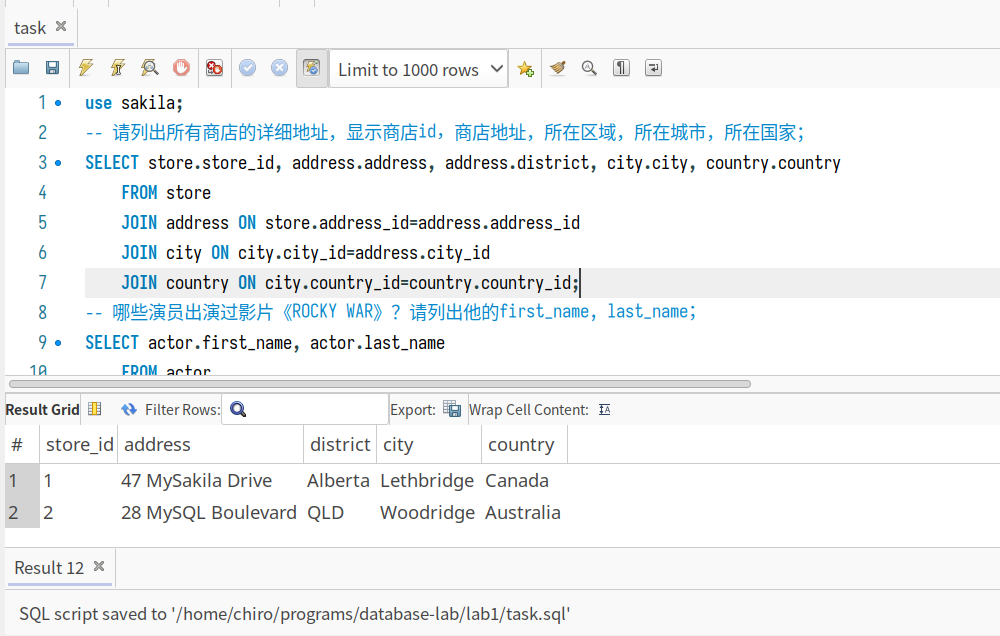


film表中引用了两个外键：language\_id、original\_language\_id，设置的外键都来自于表languague的主键language\_id

# 实验截图

*（注意截图清晰，截图时需要体现SQL语句、执行结果、Output窗口）*

1. 请列出所有商店的详细地址，显示商店id，商店地址，所在区域，所在城市，所在国家；



SELECT store.store\_id, address.address, address.district, city.city, country.country

FROM store

JOIN address ON store.address\_id=address.address\_id

JOIN city ON city.city\_id=address.city\_id

JOIN country ON city.country\_id=country.country\_id;

1. 哪些演员出演过影片《ROCKY WAR》？请列出他的first\_name, last\_name；

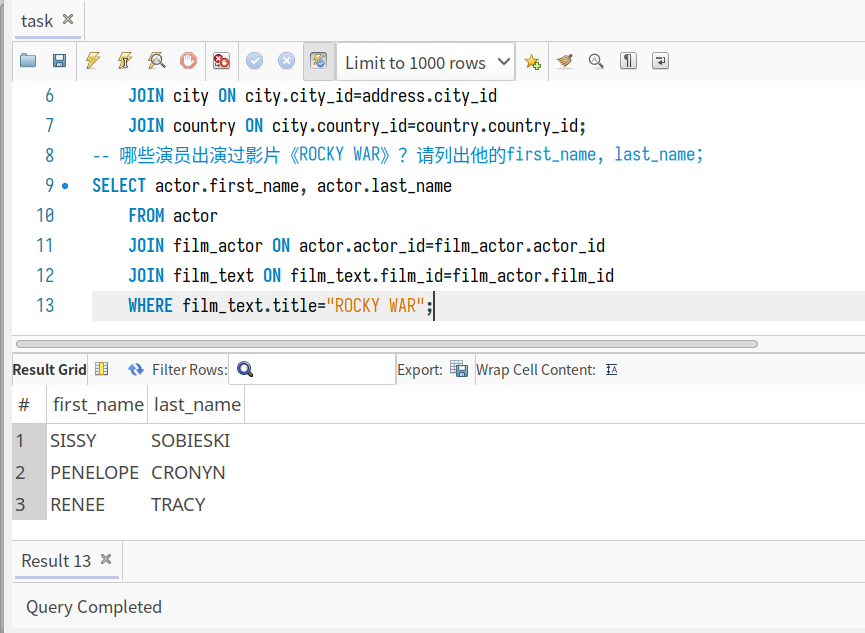
SELECT actor.first\_name, actor.last\_name

FROM actor

JOIN film\_actor ON actor.actor\_id=film\_actor.actor\_id

JOIN film\_text ON film\_text.film\_id=film\_actor.film\_id

WHERE film\_text.title="ROCKY WAR";



1. 找出租DVD花费最高的前5名，请列出他们的first\_name, last\_name和每个人花费的金额；

SELECT customer.first\_name, customer.last\_name, SUM(payment.amount)

FROM customer

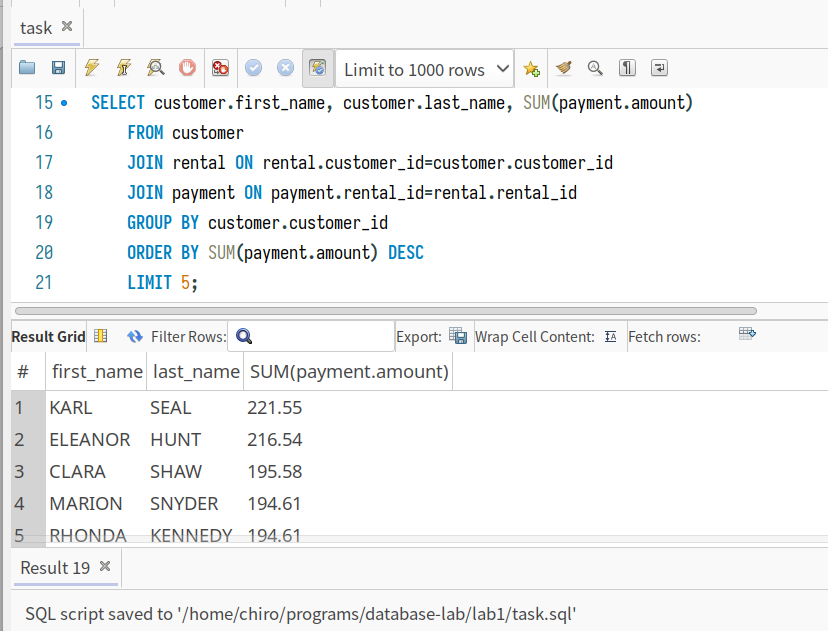
JOIN rental ON rental.customer\_id=customer.customer\_id

JOIN payment ON payment.rental\_id=rental.rental\_id

GROUP BY customer.customer\_id

ORDER BY SUM(payment.amount) DESC

LIMIT 5;



1. 哪个影片获得了总体最高的租金？请列出影片id、影片名、总租金；

SELECT film\_text.film\_id, film\_text.title, SUM(payment.amount)

FROM payment

JOIN rental ON rental.rental\_id=payment.rental\_id

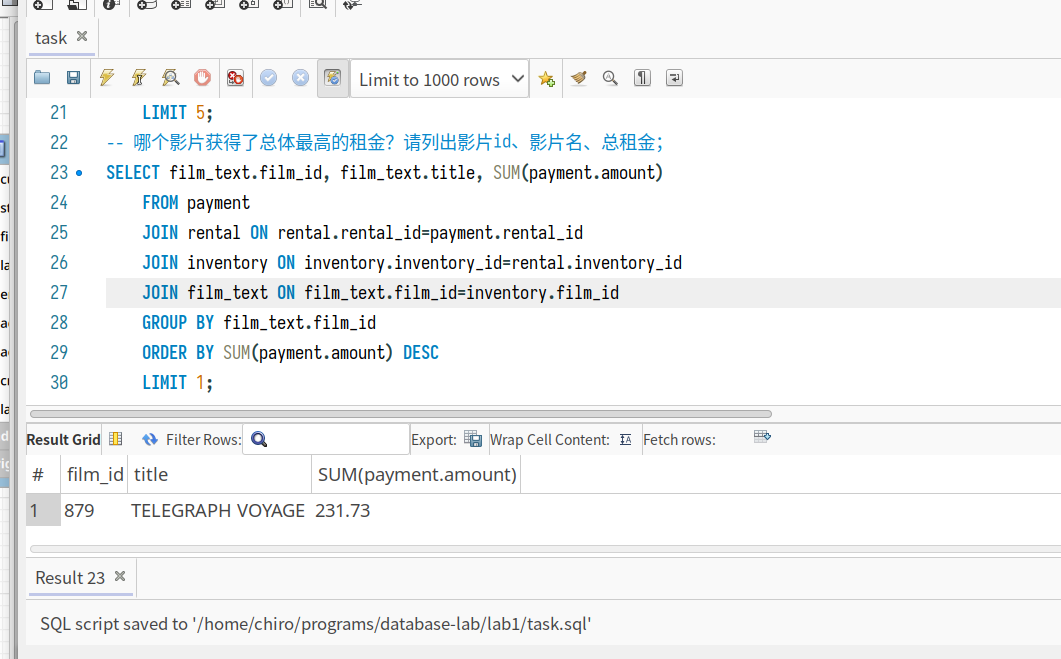
JOIN inventory ON inventory.inventory\_id=rental.inventory\_id

JOIN film\_text ON film\_text.film\_id=inventory.film\_id

GROUP BY film\_text.film\_id

ORDER BY SUM(payment.amount) DESC

LIMIT 1;



1. 哪个演员出演的电影超过35部？ 请列出演员id、演员名、出演的电影数；

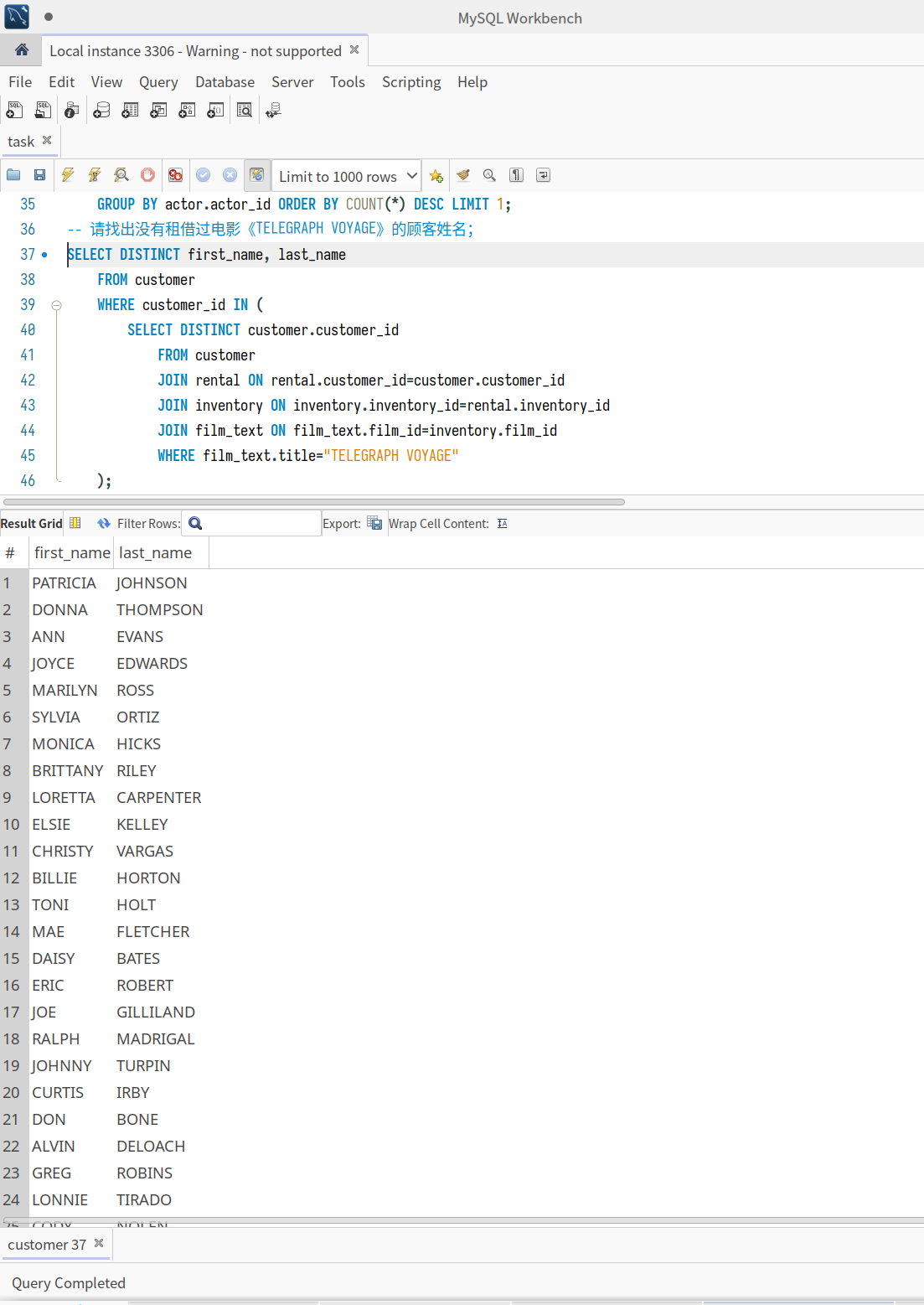
SELECT actor.actor\_id, actor.first\_name, actor.last\_name, COUNT(\*)

FROM actor

JOIN film\_actor ON actor.actor\_id=film\_actor.actor\_id

GROUP BY actor.actor\_id ORDER BY COUNT(\*) DESC LIMIT 1;





1. 请找出没有租借过电影《TELEGRAPH VOYAGE》的顾客姓名；

SELECT DISTINCT first\_name, last\_name

FROM customer

WHERE customer\_id IN (

SELECT DISTINCT customer.customer\_id

FROM customer

JOIN rental ON rental.customer\_id=customer.customer\_id

JOIN inventory ON inventory.inventory\_id=rental.inventory\_id

JOIN film\_text ON film\_text.film\_id=inventory.film\_id

WHERE film\_text.title="TELEGRAPH VOYAGE"

);

1. 查询演过《ELEPHANT TROJAN》和《SPLASH GUMP》这两部电影的演员，列出其姓名；

SELECT DISTINCT actor.first\_name, actor.last\_name

FROM actor

JOIN film\_actor ON film\_actor.actor\_id=actor.actor\_id

JOIN film\_text ON film\_actor.film\_id=film\_text.film\_id

WHERE film\_text.title="ELEPHANT TROJAN" AND actor.actor\_id IN (

SELECT actor.actor\_id

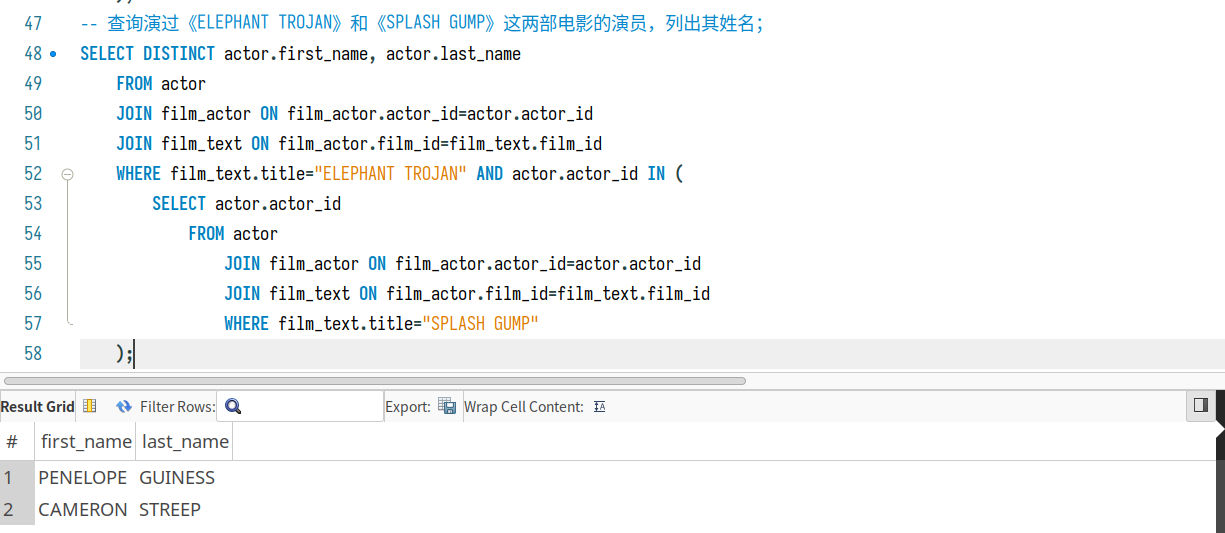
FROM actor

JOIN film\_actor ON film\_actor.actor\_id=actor.actor\_id

JOIN film\_text ON film\_actor.film\_id=film\_text.film\_id

WHERE film\_text.title="SPLASH GUMP"

);



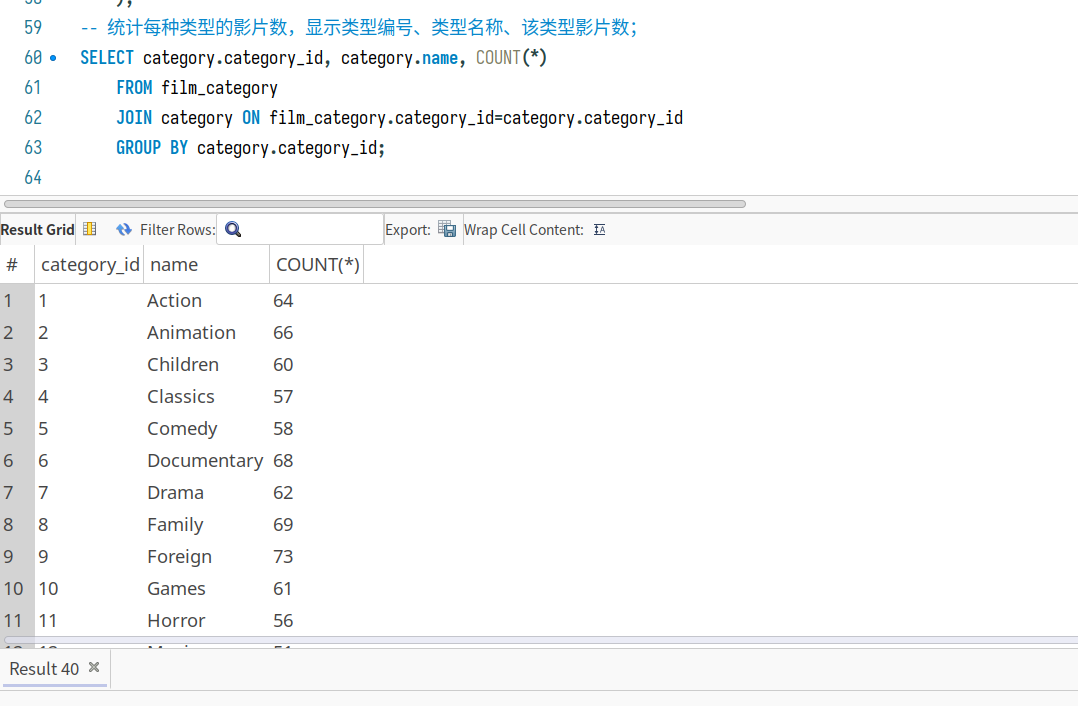
1. 统计每种类型的影片数，显示类型编号、类型名称、该类型影片数；

SELECT category.category\_id, category.name

FROM film\_category

JOIN category ON film\_category.category\_id=category.category\_id

GROUP BY category.category\_id;



1. 有哪些影片是2个商店都有库存的？

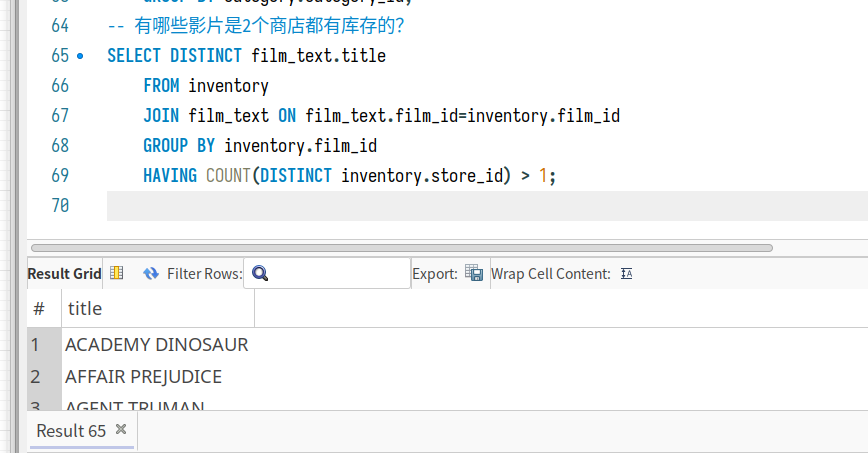
SELECT DISTINCT film\_text.title

FROM inventory

JOIN film\_text ON film\_text.film\_id=inventory.film\_id

GROUP BY inventory.film\_id

HAVING COUNT(DISTINCT inventory.store\_id) > 1;



1. 查询单次租借影片时间最长的6位客户，列出其first\_name、last\_name和当次租借时长；

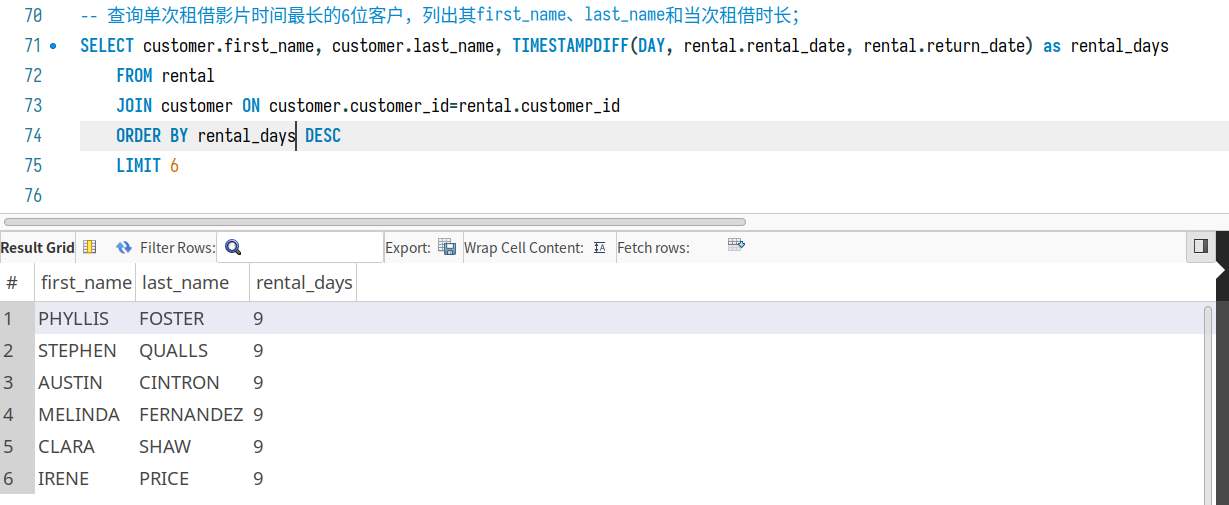
SELECT customer.first\_name, customer.last\_name, TIMESTAMPDIFF(DAY, rental.rental\_date, rental.return\_date) as rental\_days

FROM rental

JOIN customer ON customer.customer\_id=rental.customer\_id

ORDER BY rental\_days DESC

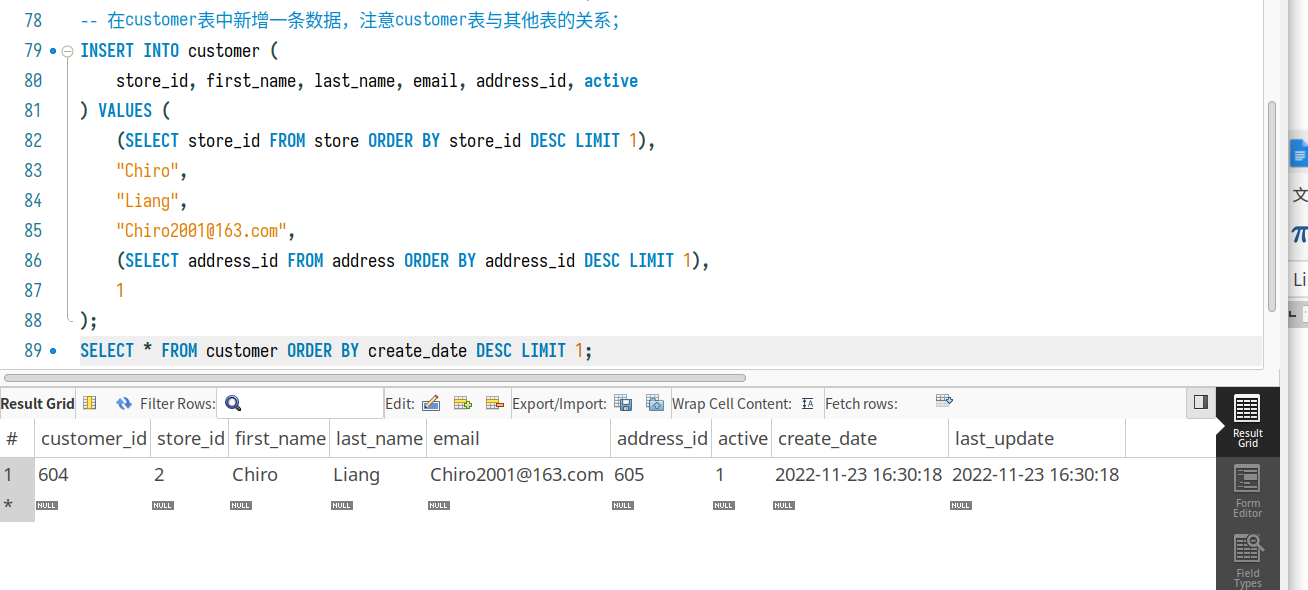
LIMIT 6;



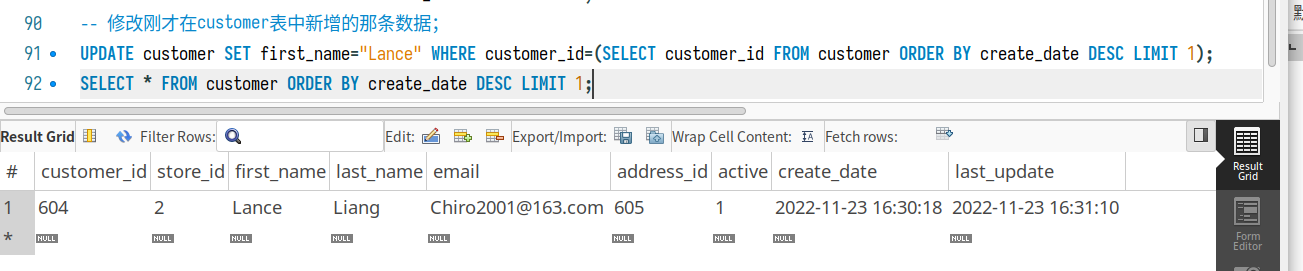
1. 在customer表中新增一条数据，注意customer表与其他表的关系；

UPDATE customer SET first\_name="Lance" WHERE customer\_id=(SELECT customer\_id FROM customer ORDER BY create\_date DESC LIMIT 1);

SELECT \* FROM customer ORDER BY create\_date DESC LIMIT 1;



1. 修改刚才在customer表中新增的那条数据；



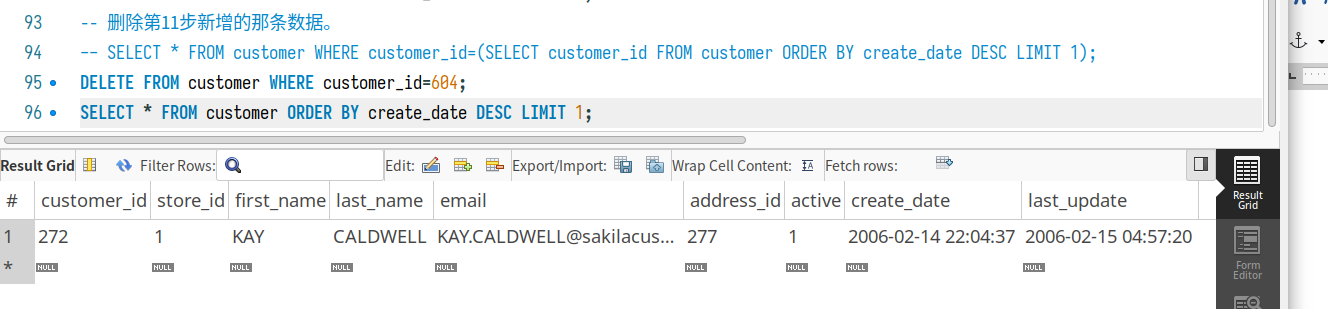
UPDATE customer SET first\_name="Lance" WHERE customer\_id=(SELECT customer\_id FROM customer ORDER BY create\_date DESC LIMIT 1);

SELECT \* FROM customer ORDER BY create\_date DESC LIMIT 1;

1. 删除第11步新增的那条数据。

DELETE FROM customer WHERE customer\_id=604;

SELECT \* FROM customer ORDER BY create\_date DESC LIMIT 1;

****

# 思考题

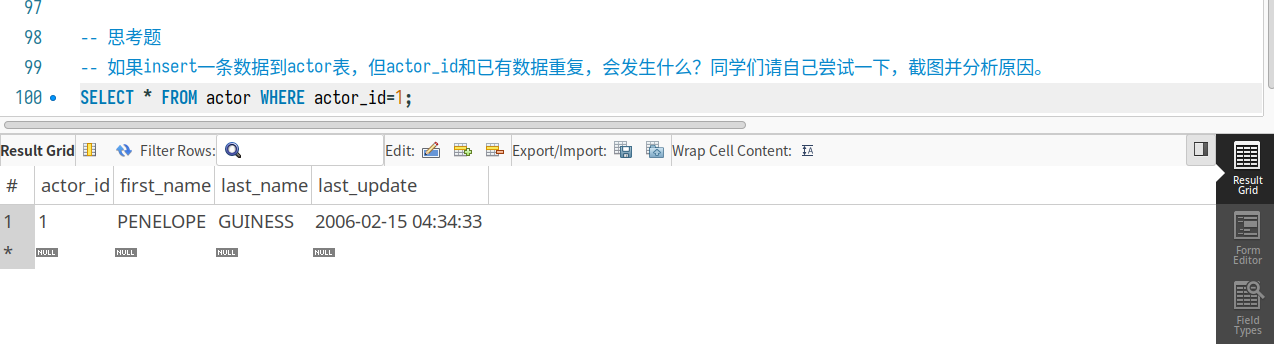
1. 如果insert一条数据到actor表，但actor\_id和已有数据重复，会发生什么？同学们请自己尝试一下，截图并分析原因。

SELECT \* FROM actor WHERE actor\_id=1;

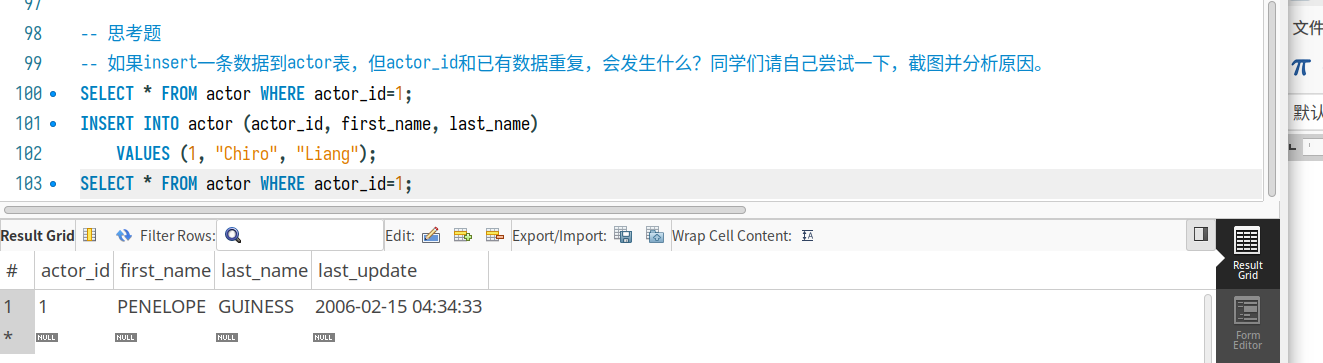
INSERT INTO actor (actor\_id, first\_name, last\_name) VALUES (1, "Chiro", "Liang");

SELECT \* FROM actor WHERE actor\_id=1;

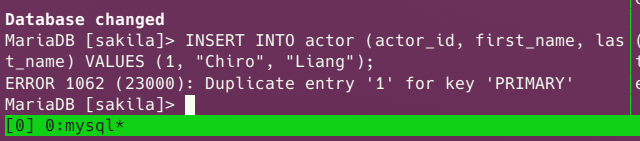
首先查询一下原数据：

在执行插入语句的时候 SQL 提示语句执行有错误，Query interrupted.

执行插入后再查询：

查询内容不变。

具体报错为：



即主键冲突时，无法插入相同主键数据。

1. insert语句还用了一个函数NOW()，是做什么的呢？

NOW() 函数返回当前日期和时间。因为表中的 create\_time和update\_time都已经配置过自动添加值，所以不使用 NOW() 函数来填充 update\_time 和create\_time 也是可以的。